System aspect relating to software development

		Standards				
Title	Objective summary	DO 178b	IEC 61508	MIL-STD-498	ISO/IEC 12207	ISO 9000-3
Scope of the Standard Interpretation of system and software	The interpretation of these terms varies with the standards.	The standard covers only the software development portion of an airborne system development.	Part I governs general development of a system comprised of electrical and/or electronic components. Part 3 covers the software development of this system.	The term system may mean a hardware/software system, for which the standard covers only the software portion or a software system for which this standard governs overall development. (1.2.4.1)	The standard covers the processes for acquiring, supplying, developing and maintaining a software or a system including software.	The standard sets out guidelines to facilitate the application of ISO 9001 to Organisations designing, developing, supplying, installing and maintaining software.
				Use of terms: acquirer, contract, developer...		
Data flow from system to software processes	Software Requirements are derived from system safety requirements within the safety assessment process.	System safety requirements are inputs to the software life cycle e.g.: criticality, software level, safety strategies and design constraints (2.1.1)	The specification of the requirements for software safety shall be derived from the specified safety requirements of the E/E/PE safety-related system and any requirements of safety planning. (7 .2.2.2)	The developer shall define and record the traceability between the CSCI (software configuration item) requirements and system requirements. (5.5)	System life cycle Processes and software life cycle processes shall be consistent. (1.2)	The standard does not address the data flow between software process and system process.
Data flow from software to system processes	The software Design implementation shall be traced to the system safety	Data includes: fault containment boundaries, software requirements, error sources detected or eliminated through software architecture. (2.1.2)	Software safety requirements shall be expressed and structured such that they are traceable back to the specification of the safety requirements of the E/E/PE safety-related system. (7 .2.2.6.b)			

Definition of	To define the System criticality categories, the software levels, their relationship and the way to derive one from the other.	The failure condition category of a system is established by determining the severity of the failure effect (functional capability , material and human consequences). This categorisation is qualitative. (2.2.1) The software levels are linked to the previous categories: they are based upon the contribution of software to failure conditions. Each software level corresponds to a failure condition category. (2.2.2) The standard provides a guidance on software level definition. (2.2.3)	The system safety assessment process identifies hazards and risks (consequences of hazard . frequency of occurrence), identifies the need for risk reduction (7.4 in part I), specifies safety functions for each hazard in order to reduce risk (7.5 in part I) and allocates each safety function to the hardware and software (7.6 in part I). A safety integrity (probability) is defined and allocated to each functions. The safety integrity may be defined qualitatively or quantitatively. Two types of probability pertaining to different modes of operation (low demand and high demand modes) are presented. (7.6.2.5 in part I). Then each failure probability corresponds to a level which is the safety integrity level of the function. (7.6.2.9 in part I). The software safety integrity level is directly derived from the safety integrity level. Requirements in the case of software functions of different safety integrity levels are given. (7 .4.2.8) The case of non-safety functions is specified (7.2.2.10 and 7.4.2.7)	The developer shall identify as safety-critical those software whose failure could lead to a hazardous system state. For such software, the developer shall develop a safety assurance strategy, including both tests and analysis, to assure that life cycle procedures minimise or eliminate the potential for hazardous conditions. The strategy shall include a software safety program and be recorded in the development plan. Evidence shall be produced, that the strategy has been carried out. (4.2.4.1)	This international standard does not address safety.	This International standard does not address safety.

Svstern aspect relating to software development

		Standards				
Title	Objectlve Summary	DO 178b	IEC 61508	MIL-STD-498	ISO/IEC 12207	ISO 9000-3
Relation Between system architecture and software levels	The aim of Architectural strategies is to limit the impact of errors and to detect them	Subsection 2.3 (and 2.2.3) provides a guidance on several architectural strategies: partitioning, multiple version dissimilar software and safety monitoring	Standard states the need for necessary risk reduction (7.5 in part I) including system architecture. No guidance is given on how to reduce risk and how system architecture could be used to decrease the risk.	The standard does not address safety.	The standard does not address safety .	The standard does not address safety
System Considerations for specific software architecture characteristics		Guidance on: user-modifiable, option- selectable, cots and field-selectable software (2.4 and 2.5)				
Interaction Between software life cycle and system verification.	This interaction should be taken into account.	System verification is not covered but system verification may provide a significant coverage of the code structure. (2.7)				

Software life cycle

		Standards				
Title	Objective summary	DO 178b	IEC61508	MIL-STD-498	ISO/IEC 12207	ISO 9000-3
equirements for re life cycle(s)	A life cycle. composed of separated but interacting processes, should be specified.	For each software a life cycle should be elaborated but neither a preferred type oflife cycle nor a organisational structure is required (section 3)	A safety lifecycle for the development shall be selected and specified during safety planning. (7.1.2.1). A lifecycle model different from the model of this standard can be elaborated provided that all the objectives and requirements are met. (7.1.2.5) The standard provides the same requirements for the overall lifecycle (7.1.4.1 in part I).	The acquirer is responsible for tailoring the standard i.e. specifying a subset of the requirements of the standard and then specifying a life cycle. (1.2.3)	The developer shall specify a life cycle model. (5.3.1.1) This standard can be tailored (removal ofprocesses) by the acquirer (Annex A). It does not prescribe a specific life cycle model (1.5). If possible, each process shall be improved (7 .3)	A software development project should be organised according to an agreed life-cycle model. Quality-related activities should be planned and implemented with respect to the nature of the life-cycle model used. This part of ISO 9000 is intended for application irrespective of the life-cycle model used. .Any description. guidance, requirement or structure of this international Standard is not intended to indicate a specific life- cycle model. (5.1)
re life cycle cesses	To list life Cycle processes	The life cycle processes are: - planning, - development (requirements, design. coding, integration) and - integral processes (verification, configuration management quality assurance, certification liaison) (3.1)	The lifecycle phases are: - requirements specification. . design and development, - integration, - validation. (clause 7) Quality and safety assurance procedures shall be integrated into lifecycle activities. (7.1.2.2) Software development planning is covered by part 1 and is included in the system safety planning.	The life cycle, called software development process in the standard, includes: - project planning. - requirements analysis -, design, - implementation, - integration, - qualification - - integral processes. (4.1) Risk management shall be performed. (5.19.1)	Life cycle processes include: - development (including requirements, design. coding. tests, integration). . integral processes (including configuration management. quality assurance, verification) (5.3). The life cycle shall be managed as specified in the management process (7.1)	
cle Process finition	The activities of each process. Their chronology and the responsibilities for them should be specified..	The life cycle activities should be specified. The sequencing of processes depends on the project and processes may be iterative. Example of life cycle of software are given including: previously developed software, partitioned function and prototyping (subsection 3.2)	Each phase shall be divided into elementary tasks with a well defined input, output and activity for each task. (7.1.2.3)	The life cycle activities may overlap. may be applied iteratively or differently to different software elements and need not be performed in a specific order. (4.1)	The activities and tasks of the development process shall be selected. These activities and tasks may overlap or interact and may be performed iteratively or recursively. (5.3.1.1)	A life cycle mode will identify a number of processes and may specify the sequence in which these processes are performed. (5.1)

Software life cycle

		Standards				
Title	Objective Summary	DO 178b	IEC 61508	MIL-STD-498	ISO/IEC 12207	ISO 9000-3
Transition criteria Between processes	Transition Criteria should be established. These criteria are used to decide to re-enter a process or to initiate a new one	Criteria depend on planning and software level. Examples are given. Feedback from other processes and partial inputs are considered. (3.3)	The standard does not allow the specification of transition criteria. Items that shall be checked during the verification of each lifecycle phase are specified in 7.9.2.6 (The transition criteria are: the verification result is positive). If a modification in an earlier safety lifecycle phase is required, then that phase and the following phases shall be repeated. (7.16.2 in part I and 7.1.2.8)	The standard does not specify the use of transition criteria.	The standard does not specify the use of transition criteria.	The standard does not specify the use of transition criteria.

Life cycle data	Data produced during the software life cycle should be defined.	Data is produced during the life cycle phase to plan, direct, explain, define record, or provide evidence of activities. Data should be unambiguous, complete, verifiable, consistent, modifiable and traceable. No specific form is required by the Standard but data form should provide for the efficient retrieval and review of data throughout the service life. Data can be placed in one or two categories related to the configuration management control placed on the data.: CCI and CC2. CC I requires more configuration Management control of the data than CC2. The minimum category assigned to each data item, and its variation by software level, is specified in Annex A. No particular data packaging method or Organisation is required. (section II) Data produced during life cycle is : Planning data (see software planning process), software requirement standards, software design standards, software code standards, software requirements data, design description, source code, executable object code, software Verification cases and procedures, Software verification results, software life cycle environment configuration index, software configuration index, problem reports, software configuration management records, software quality assurance records, software accomplishment summary.	The results of carrying out the lifecycle activities shall be documented. (7.1.2.7) The documentation shall contain sufficient information necessary for effective performance of each phase of the lifecycle and of each, planning, verification and functional safety assessment activity (5.1 in part I). Data shall be accurate, concise, understandable, consistent with the purpose, accessible and maintainable (5.2.6 in part I). All relevant documents shall be revised, amended, reviewed, approved and be under the control of an appropriate document control scheme (5.2.11 in part I). The standard provides form and structure requirements.(clause 5 in part I) Data produced during life cycle are: planning data (see safety planning), software safety requirements specification, software safety validation plan, software design description, software test specification, coding standards, source code listing, code review report, software test results, software safety validation results, software modification impact analysis results, modification log, verification report, software functional safety assessment report, software configuration management data.	The developer shall establish, control and maintain a software development library and software development files for the duration of the contract. (5.2.3 and 5.2.4) Form and content of each required document are thoroughly specified in the Data Item Description (DID). Data item are: software development plan (SDP), software test plan (STP), software Installation plan (SIP), software transition plan (STrP), operational concept description (OCD), system/subsystem specification (SSS), interface requirements specification (IRS), system, subsystem design description (SSDD), interface design description (IDD), software requirements specification (SRS), software design description (SDD), database design description (DBDD), software test description (STD), software test report (STR), software product specification (SPS), software version description (SVD), software user manual (SUM), software input/output manual (SIOM), software centre operator manual (SCOM), computer operational manual (COM), computer programming manual (CPM), firmware support manual (FSM).	This international standard is not intended to prescribe the name, format, or explicit content of the documentation to be produced. Moreover the standard does not imply that documents be developed or packaged separately or combined in some fashion. (1.5) Life cycle products (data) shall be identified in a plan. Each identified document shall be designed in accordance with applicable documentation standards. Automated documentation tools may be used. (6.1) Each process shall be documented. The documentation shall be reviewed (completeness, consistency,...). (6.4.2.7)	The supplier should establish and maintain documented procedures to control all documents and data. (6.2.1) The documents and data shall be reviewed and approved for adequacy by authorised personnel prior to issue. (6.2.3) Changes to document and data shall be reviewed and approved by the same organisations. (6.2.4)

Independence of those performing the life cycle activities	The Independence of those performing the life cycle activities shall	The independence of those performing the life cycle activities is graded against software level. Tables in annex (see tables A.I to A.I 0) specify whether each objective should be satisfied with independence or not.	The standard only specifies the Independence of those performing the functional safety assessment (FSA). The minimum level of independence of those performing the FSA is graded against the software integrity level	The standard does not address the independence of those performing the life cycle activities.	The standard specifies that those performing the verification, validation and quality assurance activities may be independent.	The standard does not address the independence of those performing the life cycle activities.

Software planning process

		Standards				
Title	Objective summary	DO 178b	IEC61508	MIL-STD-498	ISO/IEC 12207	ISO 9000.3
General requirements for the planning process	Specification of plans and standards that direct the development and the integral processes	To define the means of producing software which will satisfy the system requirements and provide confidence which is consistent with airworthiness requirements. To specify all the characteristics of the life cycle (its environment and its standards) and to make up plans (4.1) Plans and standards to be produced, and the independence of those performing the planning process are graded against software level (table A.I in annex A)	To define the management and technical activities during the lifecycle and to define the responsibilities for each phase or for activities within each phase. (6.1 part 1)	Planning is an intrinsic part of the development process, to be performed regardless of whether a deliverable is required. The developer shall develop and record plans. This planning shall be consistent with system- level planning. (5.1.1)	The planning process shall be carried out as specified in the management process (7 .1.2)	The supplier should define and document how the requirements for quality will be met.:(4.2.3.) The supplier should give consideration to the following activities: -the preparation of quality plans - the identification and acquisition of the development environment, test tools, simulation or emulation facilities, techniques, resources and skills.. - - ensuring the compatibility of the processes .the updating of verification, validation and testing techniques -the identification of suitable verification -the identification and preparation of quality records.
Planning process activities	To define the planning activities, to precise their timing and the means of executing them	Requirements for the planning activities are given. Strategies for error prevention and plan reviews should be integrated into these activities. Each characteristic of the development process should be taken into account (4.2)				

Plans		5 plans should be made up, define the transition criteria and take account of a possible plan change due to feedback. Plans to be produced are : . plan for software aspects of certification . software development plan . software verification plan - software configuration management plan – software quality assurance plan (4.3)	This process is termed the safety planning (management of functional safety in part 1 and software quality management system in part 3). (subclause 6.2 of parts 1 and 3) The functional safety planning shall define the strategy for the software procurement, development, integration, verification, validation and modification to the extent required by the SIL. (6.2.2)	The planning shall include all applicable items in the software development plan. Separate plans for quality assurance and configuration management may be developed (5.1.1). The plan provides the acquirer insight into, and a tool for monitoring, methods, activities, schedules, organisation and resources. (SDP 3.2) Plans shall be subject to acquirer approval. (5.1.6)	The supplier should develop and document the project management plan(s). This plan should specify all overall responsibilities, environments, activities, Standards, tools... (5.2.4.5) The developer should establish development plans. The plans specify standards, methods, tools, activities and responsibilities. (5.3.1.4) After coding the developer shall establish an integration plan. (5.3.8.1) Separate plans for safety requirements management can be established. (5.2.4.5.e)	Several plans should be established: quality (4.2.3.I.a and 5.5), development (5.4.2), configuration management, integration and tests plan (5.4.2.i).
. the plan for software aspect of certification	It is used by the certification authority for determining whether the proposed software life cycle complies with the software level.	It should include a system and software overview, a summary of certification basis (including software level), a summary of life cycle processes and data, the schedule and other additional considerations. (11.1)	During the safety planning, the Following should be considered: -policy, strategy, responsibilities and means - lifecycle phases to be applied -documentation structure -information extent .measures and techniques -procedures for ensuring staff competence -requirements for periodic functional	Interface with independent verification and validation agents shall be planned (SDP 10.2.5.19.5).	The standard does not require that the developer establish a plan for software aspects of certification.	This standard does not require a plan for software aspects of certification. However a quality plan should be established, reviewed, agreed and updated. It should specify all quality objectives, defined input and output criteria, responsibilities and planning. (5.5)

Software planning process

		Standards				
Title	Objective summary	DO 178b	1EC 61508	MIL-STD-498	ISO/IEC 12207	ISO 9000-3
- the software developme nt plan	It should specify everything that concems the development Processes.	It may be included in the previous plan. It should include the development standards and environment, and details for the life cycle implementation. (11.2)	-procedures for ensuring prompt follow-up and satisfactory resolution of recommendations arising from the different lifecycle phases - modification procedures throughout lifecycle - modification procedures for validated software (7 .8.2.6) -procedures for configuration management	Thorough form requirements (SDP 10.1) and thorough content requirements are provided about: processes (including a short imprecise Instruction about safety assurance), standards, environment, reviews, schedule . (SDP 10.2)	The development (engineering) planning shall be specified in the development plans.	The development plan should define how the project is to be managed, including the nature and frequency of reports to management taking into account any contractual requirements. Progress reviews should ensure effective execution of development plans (5.4.1). Subclause 5.4.2 specifies what the development plan covers.
- the software configurati on manageme nt plan	It should establish the methods to be used to achieve the objectives of the software Configuration management process throughout the life cycle.	The plan should include : - the description of the configuration management environment (procedures, methods, tools...) the activities (configuration identification, baseline and traceability, problem reporting, change control, change review, configuration status accounting, archive, retrieval and release, software load control, software life cycle environment controls, software life cycle data controls) - - transition criteria - software configuration management data - supplier control (11.4)	(corrective action to be taken shall be addressed by the software verification planning 7 .9.2.2.e) - procedures for analysing operations and maintenance performance - procedures for analysing, minimising and documenting potential hazards.	The configuration management planning is specified in the software development plan. (SDP 10.2.5.14)	The configuration management plan shall describe activities, procedures and responsibilities. (6.2.1.1)	The configuration management planning is specified in the development plan (see 5.4.2.h and 5.4.2.i)).
- the software quality assurance plan	It should establish the methods to be used to achieve the objectives of the software quality assurance process.	It may include a description of process improvement, metrics and progressive management methods. It should include the quality assurance environment (scope, organisational responsibilities and interfaces, standards, procedures, tools and methods), activities, transition criteria, timing and records definition. (II .5)	The plan for functional safety assessment shall specify: -responsibilities, competence and level of independence -resources and outputs -scope and safety bodies involved (8.2.8 in part I)	Quality assurance planning is specified in the software development plan. (SDP 10.2.5.16)	The supplier shall develop, document and update a quality assurance plan (5.2.4.5.g). The plan shall encompass activities, quality standards, contract reviews, methodology, procedures, data schedule, responsibilities and tools. (6.3.1.3)	This standard does not explicitly require a quality assurance plan, but the supplier shall establish and maintain documented procedures for planning intemal quality audits. (4.3)

- the software	It should describe the verification procedures to satisfy the verification processes objectives.	It should include the organisation, the independence methods, the verification environment, the verification methods and transition criteria of the verification processes, and considerations for Establishing independence and for specific cases (partitioning, previously developed and multiple-version dissimilar software). (11.3)	The verification shall be planned concurrently with the development, for each lifecycle phase and this information shall be documented. This planning shall refer to the criteria, techniques, tools to be used in the verification process. (7 .9.2.1 and 7.9.2.2) There is a specific chapter for safety validation planning. Safety validation covers the system-Level verification of the compliance of the system with safety requirements. (7.3)	Test planning shall Include all applicable items in the software test plan. (DID STP) Reviews are planned in the development plan (SDP 10.2.5.18)	The supplier shall develop and document a plan for the verification and validation (5.2.4.5.h). The tests shall be planned and documented concurrently with the development, for each related life cycle phase. Test planning is specified in the integration plan. (5.3.8.1) A verification (review) plan (6.4.1.5) and a validation plan (6.5.1.4) shall be established and include: life cycle activities to be verified, resources, responsibilities and	Test planning is specified in the development plan. (see 5.4.2.e and 5.4.2.i and 5.7.5.1) The verification activities should be planned and conducted in accordance with The quality plan or documented procedures to ensure that design outputs meet the design input requirements. (5.7.2.1)

Software plannlng process

		Standards				
Title	Objective summary	DO 178b	IEC 61508	MIL-STD-498	ISO/IEC 12207	ISO 9000-3
Review and assurance of the planning process	To ensure that the plans and standards comply with all requirements and that means are provided to execute them	The review and assurance of the planning process shall ensure that : - methods are compliant with the objectives -life cycle processes can be applied consistently - each process produces evidence that its outputs can be traced to their activity and inputs (subsection 4.6)	The requirements developed from the safety planning shall be formally reviewed by the organisations concerned, and agreement reached. (6.2.3 in part I)	The standard does not specify such reviews. Updates to plans shall be subject to acquirer approval. (5.1.6)	Planning requirements, selected life cycle processes, standards, procedure, environments, resources and competence shall be verified (6.4.2.2).	This international standard only addresses the development plan. The development plan should be reviewed and approved before execution. (5.4.1 and 6.5)

.SDP is an acronym that means Software Development Plan.
This plan is described in the Data Item Descriptions (DID).
The paragraph 3.2 refers to the DID section describing the SDP.

Life cycle environment

			Standards			
Title	Objective summary	DO 178b	IEC 61508	MIL-STD-498	ISO/IEC 12207	ISO 9000-3
Life cycle environment planning	The methods, tools, procedures, programming languages and hardware used to develop and verify should be defined.	To choose an environment with the aim of improving error detection, prevention and tolerance.(4.4) Considerations apply to the methods, notations, programming languages, method used for coding, software development environment tools and software verification and configuration management tools. DO 178B does not require the use of any specific method or technique. Methods, rules, constraints and tools shall be specified in terms of Development Standards. (See Software development standards)	A suitable set of integrated tools, including languages, compilers, configuration management tools, and, when applicable automatic testing tools, shall be selected for the required SIL. (7 .4.4.2) The standard requires/recommends the use of specific techniques and measures for each lifecycle phase commensurate with the SIL (see tables Ai to AiO and 81 to 89). Selecting Techniques from annexes A and 8 does not guarantee by itself that the required safety integrity will be achieved. (7. i .2.6)	This standard is not intended to specify or discourage the use of any particular software development method. The developer is responsible for selecting methods. (foreword 5) The developer shall use systematic, documented methods for all software development activities. These methods shall be described in, or referenced from, the software development plan. (4.2. i)	The life cycle environment planning shall be specified in the development plans. See also the Infrastructure process (7.2.2.i) This international standard does not prescribe a specific software development method (i.5). The developer shall choose, tailor and use standards, methods, tools and programming languages that are documented and appropriate (5.3.i.3).	The supplier should give consideration to the identification of the development environment; test tools, simulation or emulation facilities, techniques, resources and skills that may be needed to achieve the required quality. (4.2.3.2) Whether these tools and techniques are developed internally, or purchased, the supplier should validate them. (6.6)
Development environment	To establish the development environment.	Qualified tools to minimise the risk to the final software should be chosen. A verification process and standards in agreement with the software level should be developed. An error introduced by one part of the environment should be detected by another part. Specific cases are analysed: tools in combination and optional features of software tools. (4.4.i)	The design method chosen shall possess specific features. (7.4.2.2 and 7.4.2.4)	To establish, control and maintain an engineering environment in compliance with the functions to perform. (5.2.i)	The engineering environment is specified in the development plans.	See above

Language and compiler consideration	To take into account the choice of compiler and language in the software planning and verification activities.	The standard highlights the need to carefully consider the language and compiler which may impair the traceability between the source code and the object code. Planning process should provide means to ensure verification coverage and define the means in the appropriate plan. The Planning process should consider the particular features and changes of the programming language and compiler (4.4.2)	A suitable set of integrated tools, including languages and compilers shall be selected. (7 .4.4.2) Requirements for the programming language are provided (7 .4.4.3).	The standard does not provide a specific guidance for languages and compilers.	The standard does not provide a specific guidance for languages and compilers.	Tools used in the design and development, such as CASE tools, compilers, assemblers, etc. should be qualified, and placed under configuration control. Where practical, qualification should take place prior to use. (5.6.4)
Test environment	Qualified tools, methods, procedures and hardware to test the outputs of the integration process should be chosen.	Certification may be given for testing done using an emulator or a simulator. Emulator and simulator should be qualified as defined in i2.2. In case of differences between the target computer and the emulator or the simulator, the ability to detect potential errors should be considered and detection of those errors should be provided by other software verification process activities. (4.4.3 and i2.2) The test should be performed in the integrated	A suitable set of integrated tools, including when applicable automatic testing tools, shall be selected (7.4.4.2). Suitable tools are issued from tables in annex A. Methods such as probabilistic testing, dynamic analysis and testing, data recording and analysis, functional and black box testing, performance testing, interface testing, formal proof, static analysis, software complexity metrics are required for software testing for higher	To establish, control and maintain a test environment to perform qualification and possibly other testing. (5.2.2)	Methods, techniques and tools necessary to the verification process shall be chosen (6.4.1.4)	A guidance on what should be considered in establishing the test environment is provided insubclause5.7.5.i.

		target computer, since errors are only detected in it. (6.4.1)	safety integrity level. The required/recommended use of methods is indexed against the software integrity level (SIL).			

Software development standards

			Standards			
Title	Objective summary	DO 178b	IEC 61508	MIL-STD-498	ISO/IEC 12207	ISO 9000-3
Development standards	Rules and constraints for the development process and its consistency should be specified in terms of development standards.	These standards define the methods, rules and tools to be used to develop the high-Level requirements (requirements standards; 11.6), the software architecture and low-Level requirements (design standards; 11.7) and to code the software (code standards; 11.8). They are in compliance with the safety-related requirements and are a basis for the verification process. (4.5)	Coding standards shall be specified and reviewed by the assessor. (7 .4.4.5) Requirements for the specification of the coding standards are provided. (7 .4.4.6) Methods, techniques and tools concerning other processes are not specified in terms of standards (See Life cycle environment planning).	The developer shall develop and apply standards for representing requirements, design, code, test cases, test procedures and test results. These standards shall be described, or referenced from, the software development plan. (4.2.2)	The development standards shall be specified in the development plans. The adequacy of a standard shall be evaluated prior to each process using it.	This international standard does not require explicit standards but rules, practices and conventions should be specified in the development plan (5.4.2.h and 6.5). ISO 900-3 addresses elsewhere Design rules (5.6.3.a) and programming rules, languages and conventions (5.6.4).

Software development processes

		Standards				
Title	Objective summary	DO 178b	IEC 61508	MIL-STD-498	ISO/IEC 12207	ISO 9000-3
General requirements	Development activities (requirements, design, coding and integration) are applied in compliance with the planning process	The standard identifies the notion of requirements (high level requirements issued from specification, low level requirements issued from design, derived (non directly traceable to specification or design)). The document requires that standards are set and are followed. Traceability shall be ensured at all stages. Post certification modifications are not allowed.(section 5) The development activities and the independence of those performing them are graded against software level (see table *A.2* in annex I).	All development activities are performed with respect to the required safety integrity level.	The standard does not provide general requirements. See the following objectives.	This international standard describes the architecture of the software life cycle processes but does not specify the details of how to implement or perform the activities and tasks included in the processes. (1.5)	It is imperative that the design and implementation activities are carried out in a disciplined manner, in order to produce a product according to the specification rather than depending on the test and validation activities for assurance of quality (5.6.1).

Software requirements process	From the outputs of the system life cycle, this process develops the software requirements data.	Objectives are: - high-Level requirements are developed (functional, performance, interface and safety-related) -derived high-Level requirements are indicated to the system safety assessment process. Software requirements data are produced (5.1 and 11.9)	Subclause 7.2 Requirements are specified in terms of the requirements for software safety functions and the requirements for software safety integrity. (7.2.1.1) The requirements must be specified in sufficient detail to allow the Development, the functional safety assessment and the achievement of safety integrity. (7.2.2.3) Outputs shall be precise, verifiable and traceable back to the system requirements. (7 .2.2.6)	The developer shall define and record the software requirements to be met by each configuration item (CSCI), the methods to be used to ensure that the requirements have been met and the traceability between the CSCI requirements and the system requirements. (5.5) Requirements are specified in DID SRS (Software Requirements Specification).	The developer shall establish and document the software requirements (5.3.4.1). The outputs shall be traceable back to system, consistent with System requirements, verifiable (5.3.4.2).	The supplier should develop the requirements specification in close co-operation with the customer and obtain its approval. (5.3.1) The interfaces should be specified in the customer's Requirements specification. The requirements should be expressed in terms which allow validation during product acceptance. (5.3.2)

Software development processes

		Standards				
Title	Objective summary	DO 178b	IEC 61508	MIL-STD-498	ISO/IEC 12207	ISO 9000-3
Software design process	The software architecture should be defined and low-level requirements developed.	The objectives are: - software architecture and low-level requirements are derived from high-level requirements - derived low-level requirements are provided to the system safety assessment process (5.2) There is a guidance for designing for user- modifiable software. (5.2.3) Design data shall be produced. (11.10)	General requirements on design method and design implementation are given. (7.4.2) As far as practicable the design shall minimise the safety-related part of the software. (7 .4.2.6) The software architecture (7.4.3) shall be created and fulfil the software safety requirements with respect to the required SIL. The software shall be designed and implemented and be analysable, verifiable and capable of being safely modified (7 .4.5).	The process includes: CSCI (software configuration item) - wide design decisions, CSCI architectural design, CSCI detailed design. See requirements in DID SDD (Software Design Description).	The developer shall establish and document the software architectural (high-level) design (5.3.5). The developer should establish and document a detailed design of each software item (5.3.6). The outputs should be traceable back to requirements, consistent with software requirements, verifiable.	The standard provides a guidance on design (5.6.3) and addresses the use of past experiences. The design outputs should be defined and documented in accordance with the chosen methodology (5.6.5).
Coding process	From the software architecture and low-level requirements, the source code and the object code shall be developed.	Objective is : Source code is developed that is traceable, verifiable, consistent and correctly implements low level requirements. (5.3) Outputs of the process are source code (11.11) and object code.	To develop detailed code that fulfils software safety requirements with respect to the required SIL, which is readable, modifiable, understandable and testable. (7 .4.6)	Software implementation shall include coding computer instructions and data definitions, building and populating databases corresponding to each software unit. (5.7.1)	The developer shall code each software unit and database (5.3.7.1). The outputs should be traceable back to design and requirements, consistent with software design. (5.3.7.5)	The standard provides a guidance on implementation (set of activities which transfer the specified requirements to executable object code). (5.6.4)
Integration process	Executable object code is generated from the source code and loaded into the target hardware. The integration consists of software integration and system integration.	Objective of the integration process is : The executable object code is loaded into the target hardware for hardware/software integration. The integration consists of software integration and hardware/software integration. (5.4) Considerations for deactivated code and software patches are given. Evidence should be available that a deactivated code is disabled for the environments where its use is not intended. Patches are shall not be used without re-certification of the software. (5.4.3)	Software integration process is implicitly discussed in subclause 7 .4.8 (Requirements for software integration testing). System integration consists of integrating the software onto the target programmable electronic hardware (7.5.1.1).	To perform unit integration until all software in each CSCI is integrated. (5.8) To integrate CSCIs with interfacing Hardware Configuration Items (HWCIs) and CSCIs until all CSCIs and HWCIs in the system are integrated. (5.10)	The developer shall plan (5.3.8.1) and then perform unit integration (5.3.8.2). Software configuration items shall be integrated Into the system (5.3.10.1). The outputs should be traceable back to system requirements, consistent with system requirements (5.3.8.5).	

Traceability	There should be a traceability Between system specification, high- and low-level software requirements and source code.	Traceability is systematically required. It is also explicitly specified and summarised in subsection 5.5 .	Traceability is implicitly required. Software safety requirements shall be traceable back to the specification of the safety requirements of the E/E/PE system (7 .2.2.6.b).	Traceability is systematically required (also in DID). Compliance with these requirements provides a traceability from system requirements to software unit and to each test.	Traceability is systematically required.	Traceability is not addressed by the standard.
Software support	The process includes maintenance, aid to users and related activities.	Maintenance is not covered by the standard.	Maintenance shall be planned (7. 7 in part I) Chronological documentation of operation, repair and maintenance shall be maintained (7.15.2.3 in part I)	The developer shall prepare user manuals and provide assistance as specified in the contract (5.12) He shall provide information to the support	The developer shall prepare and update user manuals. (specified in each life cycle phase)	Maintenance is addressed in subclause 5.10.

Software verificatlon process

		Standards				
Title	Objective summary	DO 178b	IEC61508	MIL-STD.498	ISO/IEC 12207	ISO 9000-3
General requirements	Technical assessment of the results of both the software development and verification processes with the aim of detecting and reporting errors and with some level of Independence.	To detect and report errors that may have been introduced during the software development processes (6.1). Verification is not simply testing and includes a combination of reviews, analyses, development of test cases and procedures, execution of those test procedures (6.2). The verification activities and the independence of those performing them are graded against software level. (See tables A.3 through A.7 in annex A)	To test and evaluate the outputs from a given lifecycle phase to ensure correctness and consistency, to the extent required by safety integrity level. (7.9.1)	The standard does not provide general requirements. See the following objectives.	The verification and validation process can be performed by an independent agent as specified in the contract. The relationship with this agent is managed by the supplier (5.2.5.5)	The supplier should plan and implement verification, validation and test activities for all software developments (5.7.1). The verification results should be recorded and checked when the actions are completed. (5.7.6)
Activities	Combination of reviews, analysis, development of test cases and execution of test procedures. The verification process is performed as planned in the verification plan.	To assess accuracy, completeness and verifiability of the software requirements, architecture and source code (see reviews and analysis), and then to test the compliance with the requirements (see testing process). (subsection 6.2) Guidance on outputs is provided. (11.13 & 11.14)	To document evidence to show that each phase has been satisfactorily completed (7.9.2.4). The following should be verified (see reviews and analysis): software safety requirements (7 .9.2.8), architecture (7.9.2.9), design (system 7.9.2.10 and module 7.9.2.11), code (7.9.2.12), data (7.9.2.13). The following testing activities (see testing process) should be verified: module testing, integration testing, hardware-software integration testing and safety requirements testing (see validation).	The developer shall perform evaluations (see reviews and analysis). The processes to be evaluated and criteria to be used are given in appendix D. (5.15.1) The developer shall test the following (see testing process): -units (5.7.2), -unit integration (5.8.1) -hardware/software integration (5.10.1) The developer shall also perform a software- and system- level qualification (see validation).	Verification activities may include reviews, analysis and tests. (6.4) The verification process may be performed by an independent organisation. The verification effort shall be justified (criticality) and the level of independence of those performing the process shall be specified. (6.4.1.1) Life cycle activities to be verified shall be assessed. (6.4.1.4)	Design verification is required to be performed at appropriate points in the design process (but also at other stages in the development process, 5.7.1). It may comprise formal documented reviews of design output, demonstrations or tests. (5.7.2.1)

Verification process: reviews and analysis

		Standards				
Title	Objective summary	DO 178b	IEC61508	MIL-STD-498	ISO/IEC I2207	ISO 9000-3
General requirements	Analysis provide a repeatable evidence of correctness, reviews provide a qualitative assessment of correctness.	Subsection 6.3 provides requirements for the reviews and analysis of high level requirements, low level requirements, software architecture, software code. Outputs are recorded in the software verification results. The reviews and analysis, and the independence of those performing them are graded against software level. (See tables A.3 through A.5 in annex A)	The verifications (reviews) to be performed are specified in subclause 7.9.2.7 . The outputs to be verified are: safety requirements, architecture, System design, module design and code. The data used by software shall also be reviewed.	The developer shall perform software product evaluation: in-process evaluations and final product evaluation. The processes to be reviewed and criteria to be used are given in appendix D. (5.15.1) The developer shall prepare and maintain records of each evaluation. (5.15.2) The persons responsible for evaluating shall be independent. (5.15.3)	System criticality should be analysed to ensure that reviews are necessary. (6.4.1.I)The persons responsible for this (verification) process shall have independence and authority. (6.4.1.3) Detected non-compliance shall be provided to the problem resolution process. (6.4.1.6) All reviews aim at ensuring that outputs comply with safety , security and criticality requirements.	Most design reviews are scheduled for particular stages of the development, but may Also be unscheduled and triggered by a particular problem. The standard provides a guidance on what a review procedure should address. Records of all such reviews should be maintained. (5.7.2.2)
Review and analysis of high-level requirements	To ensure that high-level requirements comply with the system requirements.	The process shall verify that these requirements are consistent, accurate, verifiable, traceable and compatible with the target computer. They shall comply with the requirements standards. (6.3.1)	This process, termed the safety requirements verification, shall check consistency and control the validation plan. (7 .9.2.8). The developer shall review the requirements to ensure that they are adequately defined (7 .2.2.4 and 7.4.1.2) and resolve disagreements over safety integrity level. (7.2.2.5).		The requirements verification process shall verify that outputs are consistent, verifiable and traceable back to system. The process also verify that system requirements are correctly allocated. (6.4.2.3)	
Review and analysis of low-level requirements	To ensure that low-level requirements comply with the high-level requirements.	The process shall verify that these requirements are consistent, accurate, verifiable, traceable and compatible with the target computer. They shall comply with the design standards. (6.3.2)	The process, termed the system design verification, shall verify that the design is consistent with the requirements and further development, verification or modification. (7.4.1.5 and 7.9.2.10)			
Review and analysis of software architecture	To ensure that software architecture complies with the high-level requirements.	The process shall verify that architecture is consistent, verifiable and compatible with the target computer. It	The process shall verify that the architecture is consistent with the requirements and further development, verification or		The design verification process shall verify that outputs are correct, consistent, verifiable and traceable back to	

		shall comply with the design standards.(6.3.3) Guidance on partitioning integrity	modification. The verification shall control the specification of architecture integration tests. (7 .9.2.9)		system.(6.4.2.4)	

Verification process: reviews and analysis

		Standards				
Title	Objective summary	DO 178b	IEC 61508	MIL-STD-498	ISO/IEC 12207	ISO 9000-3
Review and analysis of source code	To ensure that source code complies with the low-Level requirements and the software architecture.	The process shall verify that the source code is verifiable, traceable and consistent. It shall comply with the code standards. (6.3.4)	The process, termed the module design verification, shall verify that the module design is consistent with the system design specification and further development, verification or modification (7.9.2.1 i). The code verification shall ensure conformance to the module design and the coding standards. The source code shall be verified by static methods. (7.9.2.i2)	See the previous table.	The code verification process shall verify that outputs are complete, Correct, consistent, verifiable and traceable back to system, and comply with requirements and coding standards. (6.4.2.5)	See the previous table.
Review and analysis of integration process	To ensure that the integration process results are complete and correct.	Objective may be performed by a detailed examination of the linking and loading data and memory map. (6.3.5)			The integration verification process shall verify that software items have been correctly and completely integrated in accordance with the integration plan (6.4.2.6).	
Review and analysis of the test cases, Procedures and results	To ensure that code testing was developed and performed accurately and completely.	Test cases, test procedures and test results shall be reviewed. (6.3.6)	The process is performed concurrently with the other reviews and analysis.		The standard does not require reviews of test cases, procedures and results.	
Review and analysis of the data used by software	To verify completeness, consistency, correctness, protection of data used by the software (internal data, application data, modifiable parameters).	DO 178B does not address specifically the data of the software. Guidance is given on user modifiable software (2.4, 5.2.3)	Data structures and application data shall be verified. Interfaces and associated software shall be verified. All modifiable parameters shall be verified for protection against unexpected changes. (7.9.2.i3)		The standard does not require reviews and analysis of the data used by software.	The verification results should be recorded and checked when the actions are completed. (5.7.6)
Improvement process	Reviews and analysis shall be performed with the aim of improving each process.	This document does not specifically address improvement process.	This document does not specifically address improvement process.	The developer shall periodically assess the processes used on the project to determine their suitability and effectiveness, and identify any necessary and beneficial improvements. (5. 9. 7)	Each process shall be evaluated and reviewed to identify where improvements are needed. (7.3)	

Joint review	The acquirer and the developer shall carry out reviews to assess the status and the products of a life cycle phase.	This document does not specifically address joint reviews.	This document does not specifically address joint reviews.	The developer shall plan and take part in joint (acquirer/developer) technical and management reviews (5.18)	Joint reviews shall analyse both management (6.6.2) and techniques (6.6.3). The guidance on joint reviews is quite detailed.	Regular joint reviews should be scheduled. (5.7.3)

Verification: Testing process

		Standards				
Title	Objective summary	DO 178b	1EC 61508	MIL.STD-498	ISO/IEC 12207	ISO 9000-3
System validation	To demonstrate that the integrated system conforms to the requirements specification at the intended software level.	System-level testing is not covered by the standard. It is covered in ARP 4754	The standard requires system- level tests, anyhow if the compliance with the requirements for software safety has already been established as part of the E/E/PE safety-related system, then the validation need not be repeated. The results shall be documented.(7 . 7) Validation shall be performed during actual operation.	Qualification testing aims at demonstrating the acquirer that the requirements have been met. The persons responsible for qualification testing shall be independent. The process shall include testing on the target computer. Two testing levels are required: CSCI qualification (5.9) and system qualification (5.11).	The validation effort shall be justified and the level of independence of those performing the validation shall be specified (6.5.1.1). Test cases shall be selected(6.5.2.2) and tests shall be performed (6.5.2.3).	Before offering the product for delivery and customer acceptance, the supplier should validate the operation of the product in accordance with its specified intended use, when Possible under conditions similar to the application environment. (5.7.4)
Test coverage analysis	Test coverage analysis shall be performed.	The standard requires two test coverages: -requirements-based test coverage analysis (to determine how well the testing verified the implementation of the software requirements) -structural coverage analysis (to determine how well the testing verified the code structure). They accomplish traceability between the implementation of the software requirements and their verification. (6.4.4.1 and 6.4.4.2). Case of the highest software level (6.4.4.2) and of unexecuted code (6.4.4.3).	Coverage analysis is implicitiy required by the recommended use of testing methods such as the structure-based testing method (see tables A.5, A.9 and B.2 in annex).	The test cases shall cover all aspects of the design. No coverage analysis is required.	Coverage analysis shall be carried out after each testing phase with the aim of ensuring that each compliance with requirements has been tested.	The standard does not explicitly address Coverage analysis.

Audits	Audits shall be performed to verify the compliance with requirements, plans and contract.	Quality assurance audits shall be carried out. (see quality assurance process)	Quality assurance and configuration management audits shall be carried out. (see quality assurance and configuration management processes)	Quality assurance audits and acquirer-conducted configuration audits shall be carried out. (see quality assurance process and configuration management)	Audits shall be carried out with Independence as planned. Audits shall ensure that the coded software complies with the design, that testing processes are performed accurately and completely, that the documentation complies with standards, that processes comply with requirements, and that cost and schedule are respected. (6.7) See also quality assurance process.	Quality assurance audits shall be carried out. (see quality assurance process)

Verification: Testing process

		Standards				
Title	Objective summary	DO 178b	IEC 61508	MIL-STD-498	ISO/IEC 12207	ISO 9000-3
Test case selection	To establish test cases.	Test cases should be based primarily on the software requirements. Two categories of requirements-based test cases are specified: normal range test for normal inputs and conditions, and robustness test cases for abnormal inputs and conditions. Guidance for both categories is given. (6.4.2) The testing activities and the independence of those performing them are graded against software level. (See tables A.6 and A.7 in annex A)	Test cases should be established to ensure testing of the software requirements. Tests to be applied during the three testing phases shall be specified: -module testing (7 .4.5.4), -integration testing (7.4.5.5,7.4.8.1 , 7.4.8.2) and -programmable electronics integration testing (7.5.2.1 to 7.5..2.4, 7.4.3.2.!) Methods such as probabilistic testing, dynamic analysis and testing, data recording and analysis, functional and black box testing, performance testing, interface testing, formal proof, static analysis, software complexity metrics are required for software testing for higher safety integrity level. The required/recommended use of methods is indexed against the software integrity level (SIL). (see tables in annex A and B)	The developer shall establish test cases (in terms of inputs, expected results and evaluation Criteria), procedures and data for testing: -units (5.7.2), -unit integration in CSCIs (5.8.1) -HWCI/CSCI integration (5.10.1) *HWCI: hardware configur. Item CSCI: computer soft. conI Item*	The developer shall establish and document test cases and procedures for testing: -units (5.3.7.I.b) -unit integration (software qualification) (5.3.8.4) -software/hardware integration (system qualification) (5.3.10.2)	The standard gives a guidance on what should be considered in establishing the test specifications activities: -test objective; -types of tests -test cases, data, results, criteria... The standard specifies some phases that may be tested: -software item test -integration test -system test -acceptance test (5.7.5.1)
Testing Phases	To execute the three testing phases: -module testing, -software integration testing and - hardware/software integration testing.	The requirements-based testing methods are: -hardware/software integration testing -software integration testing -low-Level testing. The hardware/software integration testing requires a specific environment or strategy. (6.4.3) Testing the executable object code is not required for the lowest software level (see table A.6 in annex A).	The three successive testing phases are: -software module testing (7.4. 7) -software integration testing (7 .4.8) -programmable electronics integration testing (7.5.2). The results shall be documented (see Data recording and analysis in table A.5 annex A).	Several phases: unit testing (5.7.3-5.7 .5), integration testing (5.8.2- 5.8.4), CSCI/HWCI integration testing (5.10.2-5.10.4) The developer shall record test and analysis results in appropriate Software Development Files (SDF).	The developer shall test: -each unit and each database (5.3.7.2) -unit integration (software qualification testing) (5.3.9) -software/hardware integration (system qualification testing (5.3.11). Test results shall be documented and reviewed for conformity to expected results. Qualification testing concludes with successful audits followed by the establishment of a baseline for design and code.	The standard provides a guidance on what should be considered when the supplier carries out testing. (5.7.5.2) The standard addresses field testing (5.7.5.3).

Software configuratlon management process

		Standards				
Title	Objective summary	DO 178b	IEC 61508	MIL-STD-498	ISO/IEC 12207	ISO 9000-3
General requirements	To define and control software configuration, manage configuration changes, control process inputs and outputs, establish baselines and aid the verification	Software configuration management (SCM) is applied in agreement with the planning process and the software configuration management plan (7 and 7.1). The SCM process includes the activities of configuration identification, change control, baseline establishment, and archiving of the software product, including the related software life cycle data. The SCM process does not stop when the product is accepted by the certification authority but continues throughout the service life. (7 .2) The depth of the SCM control (CCI or CC2) placed on the data is specified In subsection 7.3. The configuration management activities and the independence of those performing them are graded against software level (See table A.8 in annex A).	Software configuration management (SCM) should apply administrative and technical controls throughout the software safety lifecycle, in order to manage software changes and thus ensure that the specified requirements for software safety continue to be satisfied. (6.2.3.a)	5.14 The developer shall support acquirer-conducted configuration audits as specified in the contract. (5.14.2)	The configuration management process is a process for applying administrative and technical procedures throughout the software life cycle (6.2). The configuration shall be reviewed (configuration evaluation, 6.2.5.1 and documentation verification. 6.4.2. 7)	Configuration management is a management discipline that applies technical and administrative direction to the development and support life cycle of software configuration items. It is also applicable to related documentation. The CM process comprises: -configuration identification -configuration control -configuration status accounting -configuration auditing The level ofCM can be tailored to each project. (6.1.1) Only verified development outputs should be submitted to CM and accepted for subsequent use (5.7.6). Tools should be placed under configuration control prior to use. (6.6)
Configuration Identification	To label unambiguously each configuration Item and its successive versions so that a basis is established for the control and reference of configuration items.	Configuration identification should be done for life cycle data and for each configuration item. Configuration identification should be done before the use of configuration items and before implementation of change control and traceability data recording. (7.2.1)	Configuration item include at least safety analysis and requirements, specification and design documents, source code modules, test plans and results, all tools and environments. (6.2.3.c)	The developer shall identify the entities to be placed under configuration control. These entities shall include the software products to be developed or used under the contract and the elements of the software development environment. (5.14.1)	Each software item shall be identified (6.2.2.1).	The CM process comprises configuration identification (6.1.1.).
Baselines and Traceability	To define a basis for further software life cycle activity and allow reference to, control of. and traceability between configuration items.	Baselines should be established for items used for certification credit (7.2.2) A baseline for the software product should be established and defined in an index (11.16). Baseline should be protected from change.	To establish configuration baselines at appropriate points in the development and to guarantee the composition of. and the building of all baselines. (6.2.3.d)	The standard does not provide specific guidance on baselines.	Software items shall be identified in a baseline. (6.2)	The standard does not provide specific guidance on baselines.

Problem reporting, tracking and corrective action	To record and resolve process non-compliance with plan and standards. deficiencies of outputs and anomalous behaviour of products.	Problem resolution should be ensured in establishing reports. Problem reports that require corrective action of the software product or outputs of software life cycle processes should invoke the change control activity. (7.2.3) Guidance on reports is given (11.17)	After each verification, the verification documentation should include non-conformance. (7.9.2.5)	After each testing phase (5. 7.4, 5.8.3.5.9.6.5.10.3 and 5.11.6), the developer shall make necessary revisions to the software, participate in all necessary retesting, and update the appropriate software development files (SDFs).	A problem resolution process shall be performed and reviewed. It ensures that all identified problems and non-compliance are analysed and quickly resolved. and enables to understand problem	The supplier shall establish and maintain documented procedures for tracking and recording problems and implementing corrective and preventive action. (4.4 and 5.7 .5.2.c)

Software configuration management process

		Standards				
Title	Objective summary	DO 178b	IEC61508	MIL-STD-498	ISO/IEC 12207	ISO 9000-3
Change control	Recording, evaluation, resolution and approval of changes throughout the life cycle.	Configuration items and baselines are protected against changes. There should be a traceability between changes and their origin. Throughout the change activity , data affected by the change should be updated and records should be maintained. (7 .2.4)	To apply change control procedures, to document modification procedures (6.2.3.d) Modification procedures (request, impact analysis, authorisation, documentation) are specified in subclause 7.16.2 (part I).	The developer shall perform corrective action by preparing a problem/change report and implementing a corrective action system for handling each detected problem. Each problem shall be classified by category and priority (with the help of Appendix C). (5.17) Configuration control procedures establish Levels of control, persons with authority to make changes, the steps to be Followed to process change requests,(5.14.2).	The change control procedures include an analysis and an evaluation of changes, a verification and an audit. Traceability of each change shall be ensured. An audit shall be performed to control changes of software items implementing safety- critical functions. (6.2.3.1)	Areas impacted by any modifications should be identified and retested (5.7.5.2.d) Where a software item manifests a nonconformity during the development process, the investigation and resolution of such nonconformities should be controlled and recorded.(6.1.3) Analysis of the root causes of non-conformities may provide input to corrective and preventive action. (6.1.3)
Change review	To ensure that changes are assessed, approved or disapproved and to control feedback	Confirmation that affected configuration items are configuration identified. Feedback about safety-related changes is provided to the system safety assessment process. (7 .2.5)	To analyse the impact of modification, to approve or reject the request (6.2.3.d)			
Configuration status accounting	To provide the status and history of configuration items.	The objective of the status accounting activity is to provide data for the configuration management with respect to configuration identification, baselines, problem reports, and change control (7.2.6).	To document information to permit a subsequent audit (configuration status, release status...). (6.2.3.e)	Configuration status accounting is required in subclause 5.14.3 .	Configuration status accounting is required in subclause 6.2.4.1 .	The CM process comprises configuration status accounting (6.1.1).
Archive, retrieval and release	To process life cycle data so that they could be retrieved and duplicated without errors and their integrity could be ensured. To control that only authorised software is used.	Archive and retrieval activities aim at ensuring that the life cycle data associated with the software product can be retrieved in case of a need to duplicate, regenerate, retest or modify the software product. Release activities aim at ensuring that only authorised software is used. (7.2. 7)	To document the software release to permit maintenance and modification throughout the operational lifetime. (6.2.3.f)	The standard does not provide specific guidance on archives.	Original code and documentation shall be maintained. Software and documentation release shall be controlled. (6.2.6.1)	The supplier should establish and maintain documented procedures for replicating, delivering and installing the software items or products. (5.9)

Software load control	To ensure that the executable object code is loaded into the system with appropriate safeguards.	Procedures for part numbering and media identification shall be implemented Records should be kept that confirm software compatibility with the airbome system or hardware (7 .2.8) Considerations about field-loadable software are provided. (2.5)	The standard addresses installation (7.13 of part I).	The standard does not provide specific guidance on load control.	The standard does not provide specific guidance on load control. However the developer shall establish an installation plan and load the software product as specified in this plan. (5.3.12)	The standard addresses installation (5.9.5).
Software life Cycle Environment control	To ensure that the tools used to produce (develop, control, build, verify and load) the software are identified, controlled and retrievable.	Configuration identification should be established for the executable object code of the tools used to develop, control, build, verify and load the software. Control Categories CCI and CC2 apply to the Qualified tools. At least CC2 is applied to the other tools. (7 .2.9)	The guidance on configuration management does not include specific requirements for life cycle environment control.	The guidance on configuration management does not include specific requirements for life cycle environment control.	The guidance on configuration management does not include specific requirements for life cycle environment control.	The guidance on configuration management does not include specific requirements for life cycle environment control.

Software quality/safety assurance process

		Standards				
Title	Objective summary	DO 178b	IEC 61508 (. in part I)	MIL-STD-498	ISO/IEC 12207	ISO 9000-3
General requirements	To provide confidence that the software life cycle processes produce software that conforms to its requirements by assuring that these processes are performed in compliance with the approved software plans and standards. The process is applied as defined by the software quality assurance plan.	Objectives of SQA is to obtain assurance that : - Software development processes and integral processes comply with approved plans and standards. - Transition criteria are satisfied - A conformity review of the software is conducted (8.1) The SQA activities and the independence of those performing them are graded against software level (see table A.9)	The objective of the functional safety assessment (FSA) is to investigate and arrive at a judgement on the functional safety achieved by the E/E/PE safety-related systems. (8.1 .) The FSA shall be applied to all phases throughout the lifecycle (8.2.3 .) and may be carried out after each phase or after a number of phases (8.2.4 .). The minimum level of independence of those carrying out the FSA is specified in tables 4 and 5 (clause 8 .).	Software quality assurance evaluations shall assure that each activity is being Performed in accordance with the contract and the plan and that each Required product exists and has undergone evaluations, testing and corrective action. (5.16.1)	The quality assurance process should ensure that the processes (6.3.3), the products and the documentation (6.3.2) comply with requirements and plans.	The supplier shall establish and maintain documented procedures for implementing internal audits to verify whether quality activities comply with planned Arrangements and to determine the effectiveness of the quality system. (4.3) The supplier shall establish and maintain documented procedures for identification, collection, indexing, access, filing, storage, maintenance and disposition of quality records. (6.3.1) The standard addresses measurements of the quality. (6.4)
Activities	To perform audits and carry out all quality assurance procedures.	To provide assurance, with authority and independence, that plans and standards are developed and reviewed, that life cycle processes and products comply with all plans and standards by means of audits. (8.2)	Those carrying out the FSA shall have access to all persons involved in any lifecycle activity (8.2.2 .). They shall consider the activities carried out and the outputs obtained during each phase and judge the extent to which the objectives and Requirements in this standard have been met (8.2.3 .). They also consider the extent to which changes pertaining to previous recommendations of the FSA have been made (8.2.6 .). At the conclusion of the FSA, recommendations shall be produced (8.2.10 .).	The developer shall conduct on-going evaluations of software Development activities and the resulting products. (5.16.1) The developer shall prepare and maintain (for the life of the contract) records of each quality assurance activity . (5.16.2) Persons responsible for this process shall have independence, resources, organisational freedom and authority to permit objective evaluations and to initiate and verify corrective actions. (5.16.3)	The results of other integral processes (verification, validation, joint reviews or audits) can be used. Co-ordination with these processes should be ensured. Detected non- Compliance with requirements should be processed (problem resolution process). Persons responsible for this process shall have independence, resources and authority (6.3.1)	Internal quality audits shall be scheduled on the basis of the status and importance of the activity to be audited and shall be carried out by personnel independent. The results of the audits shall be recorded. Follow- up audits activities shall verify and record the implementation and effectiveness of the corrective action taken. (4.3) Quality records shall bemaintained to demonstrate conformance to specified requirements and the effective operation of the quality system (6.3.1). The standard addresses records held on electronic media (6.3.2).

Conformity revlew	To obtain assurance, prior to the delivery of software products submitted as part of a certification application, that the software life cycle processes are complete, software life cycle data are complete, and the executable object code is controlled and can be regenerated.	Activities of the review are detailed. (8.3)	The standard does not provide specific guidance on conformity review.	The standard does not provide specific guidance on conformity review.	The standard does not provide specific guidance on conformity revlew.	Before offering the product for delivery and customer acceptance, the supplier should validate the operation of the product (see validation) in accordance with its specified intended use, when possible under conditions similar to the application environment. (5.7.4)

Use of previous developed software

		Standards				
Title	Objective summary	DO 178b	IEC61508	MIL-STD-498	ISO/IEC 12207	ISO 9000-3
Use of Previous developed software	To justify and control the use of previous developed software. To assess the issues associated with the use of previously developed software including modifications, change of installation, change of application environment...	The intention to use such software is stated in the plan for software aspects of requirements. Traceability from product and data of the previous application to the new application should be ensured. In general, the impact of any modification should be assessed against the objectives of the standard. (12.1)	The previous developed software's suitability in satisfying the requirements specification shall be justified during safety planning.(7 .4.2.11) A modification of a validated software shall be initiated only on the issue of an authorised modification request under the procedures specified during safety planning. Impact on the system functional safety shall be analysed, ensuring that the software safety integrity level is sustained. (7 .8)	The developer shall identify and evaluate pre-existing developer software products for use in fulfilling the requirements of the contract and being cost-effective. The scope of the search and the evaluation criteria shall be as described in the development plan. (4.2.3.1) Appendix B is a guidance for interpreting MIL-STD-498 for incorporation of reusable software products (RSP). The RSP may be used as-is or modified and may be used to satisfy part or all the requirements.	Modification plans, procedures and reviews shall be developed and documented. The software shall be changed in compliance with the development process.	The supplier and customer should agree and document procedures for incorporating changes in a software product resulting from the need to maintain performance. A guidance on these procedures is provided. (5.9.4) Maintenance (problem Reports, change procedures, corrective action) is also addressed in subclause 5.10.

Tool Qualification

		Standards	
Title	ObjectIve summary	DO 178b	IEC 61508
General requirements	Tools should be qualified to ensure that they provide confidence at least equivalent to that of the processes eliminated, reduced or automated.	A tool may be qualified only for use on a specific system. The configuration management and quality assurance processes should apply to tools to be qualified. Tools should be qualified according to the type (development or verification tools). (12.2)	If tools are used as part of design or assessment for any overall, E/E/PES and software safety lifecycle activity, they should themselves be subject to the functional safety assessment. The degree to which the use of tools will need to be evaluated will depend upon their impact on the functional safety of the system. (8.2.5 in part I)
Qualification criteria for development tools	Development tools can introduce errors, therefore, stringent criteria shall be applied to their qualification.	Qualification of a tool is needed when processes are eliminated, reduced or automated by the use of a tool without its output being verified. If a tool is to be qualified, the tool should satisfy the same objectives as the software it produces. The software level assigned to the tool should be the same as that for the software it produces, unless the applicant can justify a reduction in software level of the tool to the certification authority. The applicant should prove and verify that the tool complies with its tool operational requirements. (12.2.1)	The standard only addresses programming languages: the programming language selected shall have a translator/compiler which has either a certificate of validation to a recognised national or international standard, or it shall be assessed to establish its fitness for purpose. (7.4.4.3)
Qualification criteria	The tool to be qualified should satisfy less	The qualification criteria for software verification tools	The standard only addresses verification tools

for verification tools	stringent criteria because a verification tool cannot introduce errors, but may fail to detect them.	should be achieved by demonstration that the tool complies with its tool operational requirements under normal operational conditions. (12.2.2)	used during system validation: validation tools shall be qualified according to a specification traceable to a recognised standard. (7.7.2.7)
Qualification data	The tool qualification process and data shall be described in a document.	The data are the tool operational requirements (which satisfy the same objectives as the software requirements data and describe the tool operational functionality). For a development tool, there are also a qualification plan (which satisfies the same objectives as the plan for software aspects of certification and describes the qualification process) and a tool accomplishment summary (which satisfies the same objectives as the software accomplishment summary). (12.2.3)	No specific documentation is required for qualification data.

Use of specific methods and techniques

	Standards	
Title	DO 178b	IEC 61508
Use of specific methods and techniques	An alternative method (for example: formal methods, exhaustive input testing, product service history.) is not recommended and should satisfy the objectives of the standard. The effort for obtaining certification credit of an alternative method is dependent on the software level and on its impact on the life cycle. The applicant should justify the use of these methods. (12.3) A guidance on method examples is provided. (12.3.1 12.3.2 and 12.3.5) A specific verification process should be implemented for multiple-version dissimilar software. (12.3.3)	The standard requires/recommends the use of specific techniques and methods at any stage of the lifecycle commensurate with the SIL (see tables AI to A 10 and 81 to 89). Among these required methods, alternative methods such as formal methods are recommended for specific lifecycle phases.

Software Certification

		Standards				
Title	Objective summary	DO 178b	IEC61508	MIL-STD-498	ISO/IEC 12207	ISO 9000.3
General requirements	Legal recognition by the certification authority that the software complies with the requirements	The certification authority considers the software as a part of the system and does not approve it as a stand-alone product. (section 10)	The certification process is not covered by the standard as such.	Independent verification and validation (IV&V) is not within the scope of this standard. (3.23)	Acceptance (5.1.5) To conform to the standard, the developer shall implement all the life cycle processes as specified in the contract. (1.4)	When the supplier is ready to deliver the validated product, the customer should judge whether or not the product is acceptable. (5.8.1)
Certification authority	This organisation carries out the certification process.	The certification authority is an organisation or person responsible within the state or country concerned with the certification of compliancewith the requirements. (Glossary)	Anyhow the functional safety assessment (FSA) may be considered as a mean of co-ordination with the certification authority .	The developer shall interface with the IV&V agent(s) as specified in the contract. (5.19.5)	The acquirer prepares the acceptance process and specifies the participation of the supplier (developer). Acceptance criteria shall be established. (5.1.5.1)	Acceptance tests should be performed by the customer or may be performed on behalf of the customer by the supplier or a third party. (5.8.1)
Certification Planning	To establish communication and understanding between the applicant and the certification authority throughout the software life cycle to assist the certification process	The process is applied as defined by the planning process and the plan for software aspects of certification. (section 9) Certification activities and the independence of those performing them are graded against software level (see table A 10 in annex A).		The developer shall use software management indicators to aid in managing the development process and communicating its status to the acquirer. (See Appendix F) (5.19.2)		The supplier should assist the customer to establish the acceptance test activities. (5.8.2)

Data Submitted to The certification authority	To obtain agreement with the certification authority on this plan	To submit the plan and other requested data to the certification authority for review at a point in time when the effects of changes are minimal, to resolve issues identified by the certification authority. (9.1)		The developer shall provide the acquirer access to developer and subcontractor facilities, including the software engineering and test environments.	The standard does not address the data submitted to the certification authority .	The standard does Not address the data submitted to the certification authority.
Certification process	To provide evidence that the life cycle processes satisfy the plans	To arrange review of the life cycle processes, to submit the software accomplishment summary, the configuration index and other requested data, to resolve issues raised by the certification authority as a result of its reviews. (9.2)		The standard does not address the certification activities.	The acquirer performs acceptance tests and reviews (5.1.5.2) with the help of the developer (5.3.13).	The supplier should carry out acceptance tests. (5.8.2)